Using Star Clusters for Filtering

Javed Aslam Katya Pelekhov Daniela Rus

Department of Computer Science
Dartmouth College
Hanover, NH 03755

Abstract

We examine applications of clustering to the filtering task. We use the on-line version of the star algorithm [JPR98, JPR99] as the clustering tool because this algorithm computes, with high precision, naturally occuring topics in a collection and it admits an efficient on-line solution for dynamic corpora. We describe several filtering algorithms and show extensive experimental results using the TREC collection.

1 Introduction

Our goal is to automate information access and organization by topic. In our previous work [JPR99, JPR98], we presented an efficient algorithm for organizing static and dynamic information by topic using the star cluster algorithm. We do not impose the number of topics for the collection; that is, the algorithm discovers as many topics as there are present in the collection. Furthermore, our probabilistic analysis shows guarantees on the accuracy of each topic and on the time performance of the algorithm. In a more recent paper [JRR00] we present precision-recall and performance results for the star information algorithm with sampling. In this paper we examine applications of the star information organization algorithm to the filtering task.

In the filtering application, the input is a stream of documents. The user marks documents of interest and requests similar texts from the stream of documents. In this paper we propose keeping the documents that have already been examined organized in clusters corresponding to topics. We believe that this improves the performance of the filtering system. In a clustered collection each new document can be compared to the previously relevant documents, not just a query, which can potentially improve the filtering results. The filtering system presents the user not with the clusters, but with separate documents, and decides automatically which clusters to track. Cluster quality and the ability to update clusters incrementally are extremely important in this application.

We present algorithms for this approach to filtering that use the star clustering algorithm, and experimental data. Our experiments provide further evidence that the star clustering algorithm is effective for information access in dynamic corpora. Because the on-line version of the algorithm is fast, relatively simple, and has good precision, the method is especially suitable to information retrieval tasks that have dynamic input streams, such as filtering.

2 Related Work

The filtering task, defined as the problem of selecting relevant examples from an incoming stream of documents based on an initial filtering profile and occasional relevance feedback from a user, is the subject of study in the filtering track of the Text REtrieval Conference [Hul97]. The classic filtering problem is characterized by the stability of the user's interest. The TREC study formalizes the filtering task, supplies test collections, provides valuable discussion of evaluation methods and comparison of approaches. Based on the amount of training information available, the filtering task is divided into subtasks: adaptive filtering and batch filtering.

Batch filtering operates with predefined topics and extensive relevance information. Schütze *et al.* [SHP95] demonstrated that statistical classification algorithms outperform the popular Rocchio relevance feedback algorithm in learning topic profiles for document routing. In addition, Hull *et al.* [HPS96] noticed that the combination of classifiers has better performance than individual algorithms.

Adaptive filtering operates under a realistic assumption that relevance feedback could be obtained only for the documents already retrieved. This setup requires learning algorithms to adjust to the minimal amount of training data, and to solve the problem of efficient selection of learning examples. Callan [Cal98] proposed solutions to some of these problems: an incremental algorithm for learning filtering profiles responsive to new relevance feedback, and an algorithm for learning dissemination thresholds that could be adjusted to user preference with respect to precision and recall. The clustering scheme used by University of Iowa team in TREC-7 [ERS98] employs two-level single pass clustering based on a number of empirically obtained thresholds to weed out non-relevant documents based on their cluster membership and previously obtained relevancy information. Mostafa *et al.* [MMLP97] describe a general model for information filtering systems that identifies relevance feedback and changing user interests as two issues important to filtering models. Their model is based on a two-level decomposition of the filtering function: the first level maps the documents into a finite number of classes, the second level function estimates user relevance for the different classes. They argue that the model leads to increased system efficiency, and reduced complexity.

3 Background

Systems like Scatter/Gather [CKP93] provide a mechanism for user-driven organization of data in a fixed number of clusters, but the users need to be in the loop and the computed clusters do not have accuracy guarantees. Scatter/Gather uses fractionation to compute nearest-neighbor clusters. Charika et al. [CCFM97] consider a dynamic clustering algorithm to partition a collection of text documents into a *fixed number* of clusters. Since in dynamic information systems the number of topics is not known *a priori*, a fixed number of clusters cannot generate a natural partition of the information.

To compute accurate topic clusters, one possibility is to formalize clustering as covering similarity graphs by cliques. A clique cover will guarantee that its documents are strongly related to each other. Covering by cliques is intractable. We instead propose using a cover by *dense subgraphs* that are *star-shaped* and that can be computed *off-line* for static data and *on-line* for dynamic data. What we lose in intra-cluster similarity guarantees, we gain in computational efficiency.

We represent the document collection as a complete similarity graph, where the vertices correspond to documents and the edges are weighted by a similarity measure. We have used two measures: the cosine metric and an information-theoretic metric. To compute accurate topic clusters, we computed a thresholded similarity graph, there the thresholding parameter σ is given by the smallest similarity we would like to have between any documents within a topic. We then approximate a clique cover of this graph by covering the associated thresholded similarity graph with *star-shaped subgraphs*. A star-shaped subgraph on $m+1$ vertices consists of a single *star center* and m *satellite vertices*, where there exist edges between the star center and each of the satellite vertices. A greedy algorithm computes this cover for static collections (see Figure 1).

In [JPR98, JPR99] we show an on-line version of this algorithm that supports information organization in dynamic collection. The intuition behind the incremental computation of the star cover of a graph after a new vertex is inserted is as follows. Suppose a new vertex is inserted in the star cover (which corresponds to the addition of a new document. How does the addition of this new vertex affect the correctness of the star cover? In general, the answer depends on the degree of the new vertex and on its adjacency list. If the adjacency list of the new vertex does not contain any star centers, the new vertex can be added in the star cover as a star center. If the adjacency list of the new vertex contains any center vertex c whose degree is equal or higher, the new vertex becomes a satellite vertex of c. The difficult cases that destroy the correctness of the star cover are (1) when the new vertex is adjacent to a collection of star centers, each of whose degree is lower than that of the new vertex; and (2) when the new vertex increases the degree of an adjacent satellite vertex beyond the degree of its associated star center. In these situations, the star structure already in place has to be modified; existing stars must be broken. The satellite vertices of these broken stars must be re-evaluated.

For any threshold σ:

1. Let $G_\sigma = (V, E_\sigma)$ where $E_\sigma = \{e : w(e) \geq \sigma\}$.
2. Let each vertex in G_σ initially be *unmarked*.
3. Calculate the degree of each vertex $v \in V$.
4. Let the highest degree unmarked vertex be a star center, and construct a cluster from the star center and its associated satellite vertices. Mark each node in the newly constructed cluster.
5. Repeat step 4 until all nodes are marked.
6. Represent each cluster by the document corresponding to its associated star center.

Figure 1: The star algorithm

Star-graph covers are interesting because they provide accuracy guarantees on the computed topics. By investigating the geometry of the problem, we can derive a *lower bound* on the similarity between satellite vertices as well as provide a formula ($\cos\gamma \geq \cos\alpha_1 \cos\alpha_2 + \frac{\sigma}{1+\sigma}\sin\alpha_1 \sin\alpha_2$, where α_1 and α_2 correspond to the similarity between the center and the two satellites and σ is the similarity threshold) for the *expected* similarity between satellite vertices using the cosine metric. This formula predicts that the pairwise similarity between satellite vertices in a star-shaped subgraph is high, and together with empirical evidence supporting this formula [JPR98] we conclude that covering thresholded similarity graphs with star-shaped subgraphs captures the topic contents of the document collection.

In [JPR99, JPR98] we evaluated extensively the performance of this method on several large collections. We found that the star algorithm represents a 16.7% improvement in performance with respect to average-link and an 40% improvement with respect to single-link.

3.1 An Information Organization System

We have built an information organization system and presented the performance results for this system in [JPR99, JPR98]. We have extended this system to support filtering, as described in this paper. Figure 2 shows the interface to our automated information organization on a digital library composed of news articles related to terrorism. A user has several choices at the top level:

1. *Add from Web* allows the user to specify a set of URLs. A Web Crawler is then synthesized to travel to the given URLs, bring back the file and create a directory with these files. A reference index is then synthesized for this specified collection.
2. *Add Websearch* allows the user to specify a search operation by keywords. The user types the keywords,

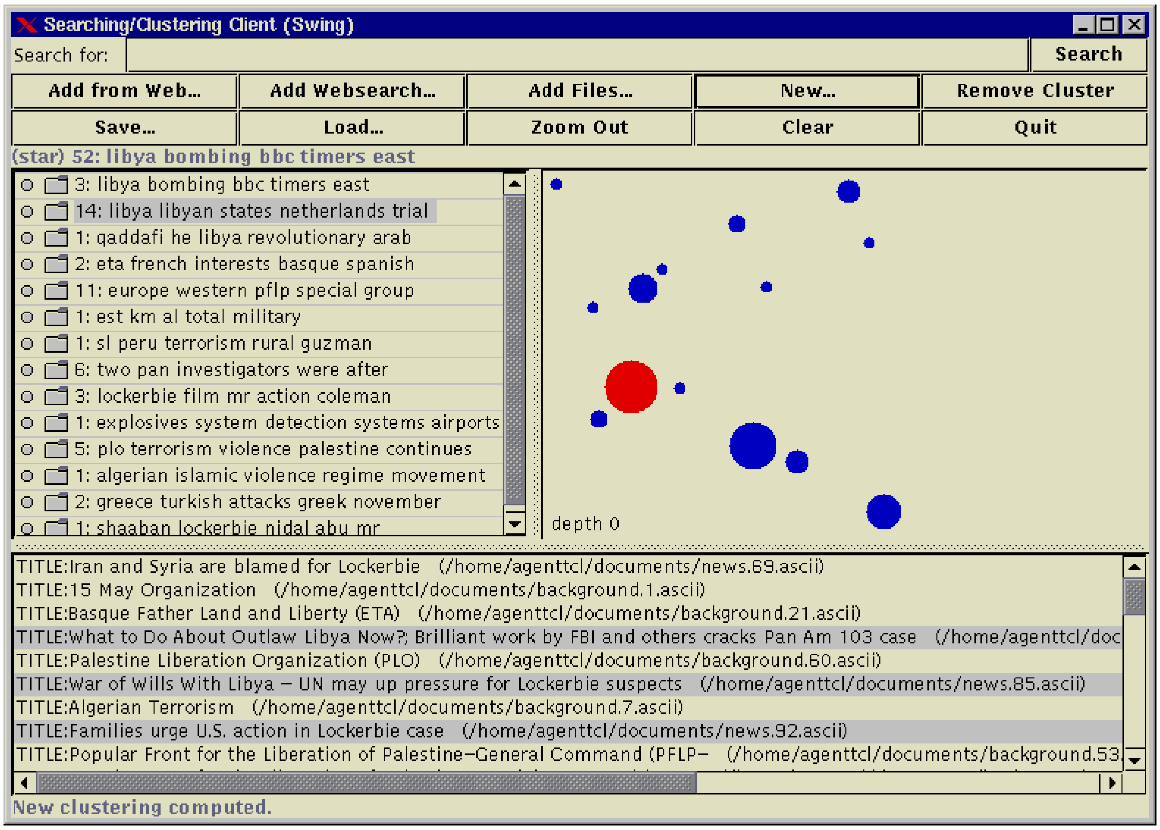

Figure 2: This figure shows the interface to the information organization system. The search and organization choices are described at the top. The middle two windows show two views of the organized documents retrieved from the Web or from the database. The left window shows the list of topics, the number of documents in each topic, and a keyword summary for each topic. The right window shows a graphical description of the topics. Each topic corresponds to a disk. The size of the disk is proportional to the number of documents in the topic cluster and the distance between two disks is proportional to the topic similarity between the corresponding topics. The bottom window shows a list of titles for the documents. The three views are connected: a click in one window causes the corresponding information to be highlighted in the other two windows.

which are then passed to a search engine. The results of the search are retrieved by the Web crawler, stored in a directory, and organized automatically.

3. *New* allows the user to specify a new digital library. The system hierarchically organizes the directory by topic and presents this information to the user. The results from *Add from Web* and *Add Websearch* can be piped through this command.

4. *Save* allows the user to save an organization digital collection for future reference.

5. *Add Files* allows the user to add new digital documents to an organized digital collection. This operation is efficient, as the files are incorporated in the collection incrementally by the on-line star clustering algorithm.

6. *Remove Cluster* allows the user to remove the documents corresponding to a topic from an organized collection. This operation is also efficient using the on-line star clustering algorithm.

7. *Zoom In* allows the user to traverse a hierarchical organization structure downwards. The user zooms in by clicking. This operation displays the subtopics contained within the topic cluster.

8. *Zoom Out* allows the user to traverse a hierarchical organization structure upwards.

Our visualization and interaction user interface is based on three views of the digital collection (see Figure 2):

1. The top-left window shows the topics in the collection by detailing their size and keyword summaries. The keywords that summarize and differentiate the topics are computed automatically and depend only on the contents in the collection. Our automatic keyword construction algorithm extracts (1) the keywords that summarize each level in a hierarchy of

topics and (2) the keywords that differentiate between different topics at the same level. The user may click once on a folder to select a particular topic. This operation selects the corresponding data in the other two views. By clicking twice, the user expands the topic to the next level in the hierarchy.

2. The top-right window presents a visualization method for large-scale information we have developed [JPR99, JPR98]. The topics that occur in the collection are shown as disks whose size is proportional to the size of the corresponding cluster. The distance between two clusters is proportional to the distance between the corresponding disks. Thus, if the user finds a topic of particular interest, this visualization suggests neighboring topics that might be related. However, such an arrangement of disks may not be possible in only two dimensions, so an arrangement which *approximately* preserves distance relationships is required.

   The user can may click on a disk once to highlight it. This operation also highlights the corresponding data in the other three views. The user may enter a topic and proceed to the next level in the hierarchy by clicking twice.

3. The bottom window lists the titles of the documents in the collection. The user may click on a document once to highlight it. This operation highlights the clusters to which the document belongs in the other two views. The user may click twice to view the text of the document.

4 Filtering

Filtering is the task of selecting documents relevant to a query from an incoming stream. Typically, the filtering system decides whether a new document is relevant instantly, without waiting for the subsequent documents to arrive. The user may correct the *filtering profile* by providing relevance feedback on the retrieved documents.

4.1 Filtering algorithms

We use the topic clusters computed with the star algorithm (described in Section 3) and relevance feedback information to compute the relevancy of a new document. Our clustering approach to document filtering is based on the premise that the similarity between a new document and a star center that corresponds to a given topic approximates well the relevance of the document to the topic. We base this assertion on the following observations:

- the cluster hypothesis ("closely related documents are relevant to the same queries");
- the star clustering algorithm finds accurate clusters, as shown in [JPR99, JPR98]; and
- a cluster obtained with the star clustering algorithm is well-represented by the document at the star center.

Thus, we define the following rules for determining the relevancy of a document based on the relevancy the cluster center:

1. A document is relevant if its adjacent center is relevant.
2. A document is not relevant if its adjacent center is not relevant.

These rules do not address the case when the star clustering algorithm places a new document in more that one cluster, or at the center of a cluster. These cases are resolved depending on the user's goals and the design of the filtering system. The following are the examples of possible strategies.

1. If a document is adjacent to both relevant and non-relevant centers, we may consider it to be
   - *relevant* if recall, or fraction of all relevant documents that are retrieved, is important;
   - *non-relevant* if precision, or fraction of relevant documents in the retrieved set, is valued.
2. If a document is a cluster center we may
   - decide document relevancy based on the relevancy of other documents in the cluster (*e.g.*, if at least 90% of documents in the cluster are relevant, then the center is relevant as well);
   - ask the user to provide input.

Given the rules for deciding the relevancy of a document to the filtering profile based on the previous observations, we formalize our approach to document filtering:

1. Select clustering threshold.
2. Cluster documents using the star algorithm.
3. Obtain initial relevancy information.
4. For each new document:
5. Add the document to the clustering using the star algorithm Update procedure [JPR99].
6. Decide whether the document is relevant based on its cluster membership.
7. Retrieve the document, if relevant; correct relevance information based on the user's input.

An implementation of a filtering system based in this algorithm needs to address the following points:

- Which threshold parameter for clustering should be selected?
- How is the relevance feedback obtained?
- How should we resolve undefined cases when deciding document relevancy?

We answer these questions, present an implementation of our filtering system, and explore its performance in the next sections.

4.2 The Filtering Algorithm

The filtering algorithm described in this section assumes the following *minimal* functionality of a filtering system. The user supplies the topic description in the form of keywords, or as a sample set of relevant documents. As new documents arrive, they are compared to the *filtering profile* (topic description), and the documents matching the topic are presented to the user. The user may correct the filtering profile by providing relevance feedback on the *retrieved* documents. The retrieved documents are ordered only by the time of their arrival. It is safe to assume that the user would prefer the retrieval of a smaller set of documents most of which are relevant over a more comprehensive, but less precise list. The details of the filtering algorithm used by this system are not available to the user. The system selects an appropriate clustering threshold based on the filtering profile and relevance information (described in Section 4.3.3).

We developed an algorithm for deciding document relevancy that emphasizes precision of the retrieved set. This algorithm is based on the update procedure of the on-line version of the star clustering algorithm [JPR99]. A new document is added to the clustering using the star algorithm update procedure. If the new document is not at the center of a new cluster, then it is relevant if all its adjacent cluster centers are relevant. If the new document is at the cluster center, it is relevant if at least 75% of its adjacent vertices are relevant.

This procedure is the basis for building our filtering algorithm. We observe that only the documents in the vicinity of the current topic profile affect the clustering of relevant documents. Pre-filtering the documents around the topic profile will increase system's efficiency, without decline in performance. The resulting filtering algorithm is as follows:

1. Find pre-filtering threshold θ.
2. Cluster pre-filtered set using the star algorithm.
3. Select clustering threshold σ, based on the topic description and the initial relevant document set.
4. For each new document α within distance θ from the filtering profile:

   Add the document α to the clustering using the procedure described above.

   If α's relevance tag is *true*, retrieve the document α, and correct its relevancy, if necessary.

We will select the pre-filtering and clustering thresholds to optimize the filtering performance as shown in Section 4.3.3. We evaluate this filtering algorithm in the next section.

4.3 Filtering Evaluation

In this section, we describe the *document collections* used to evaluate our filtering system, the *evaluation metrics* used to assess performance, and the results of our *experiments*.

4.3.1 Collections

In our experiments we used two collections consisting of news articles that were used in TREC, and a smaller collection of news concentrating on recent significant world events.

FBIS This collection used in the TREC-6 filtering study consists of 130,471 training and 120,653 test documents, with relevance judgments available for 47 topics.

AP This is a filtering collection from TREC-7 that consists of training documents from the AP88 collection, test documents from the AP89 and AP90 collections, and 50 topics.

News Documents from our news collection are a part ClariNews distribution. The collection consists of 9301 documents from the following `clari` groups: `hot.n` (special group for the impeachment of President Clinton), `hot.o` (special group for the conflict in Iraq), `hot.a` (special group for the crisis in Kosovo), `usa.military`, `usa.top`, `world.americas.south` (parent group for `world.americas.brazil`), `world.europe.russia` and `world.top`. Four groups were selected to represent filtering topics. The groups `clari.hot.n`, `clari.hot.o`, `clari.hot.a` and `clari.world.americas.brazil` were chosen for their concentration on well-defined themes (impeachment, Iraq, Yugoslavia, and the Brazilian economy). The topic size are 441, 861, 3586 and 266 documents, respectively. The collection was split into training and testing halves so that the topics are represented equally in both parts, and the documents in the training half of a topic precede those in the testing part.

4.3.2 Evaluation measures

The TREC filtering track experimented with different evaluation measures over the years. For administrative and practical reasons, the TREC filtering task uses utility measures F1 and F3 (see Table 1). Precision and recall based measures were discarded because of their inability to differentiate between systems that return no relevant documents (where clearly returning no documents at all is a much better behavior than returning a great number of non-relevant documents).

	Relevant	Not Relevant
Retrieved	R^+	N^+
Not Retrieved	R^-	N^-

$$F1 = 3R^+ - 2N^+$$
$$F3 = 4R^+ - N^+$$

Table 1: Utility measures.

The drawback of these utility measures is that they cannot be meaningfully averaged over all topics. Furthermore, these (and other) utility measures place equal value on all relevant documents. An example cited in the TREC-7 filtering report [Hul98] points out that the 1000th retrieved relevant document will likely provide no new content over the previous 999 relevant documents, and should therefore be valued less than the others. One can similarly argue about the relative importance of the 998th relevant document, and so on. It is not only the fact that we already have many relevant documents that makes a new relevant document potentially uninteresting, but the actual *content similarity* of this document to the previous ones.[1] Unfor-

[1] In fact, a filtering system that uses clustered organization is well suited to dealing with this problem by separating the documents with new content from the rest. This way, the total number of

tunately, current performance measures do not take content retrieval into account. Given a choice of traditional performance measures, we select one such measure:

$$F = \frac{2pr}{p+r}\text{[Rij79, Sha86]},$$

where p is precision and r is recall for our experiments. Our preference for this measure is based on the ease of averaging and comparison across the topics, even though we are aware of the limitations of recall-precision based evaluation measures.

4.3.3 Threshold selection in the filtering algorithm.

The following experiment was used to determine the empirical clustering threshold used in our system. We used the FBIS collection where we kept the most represented topics (a total of 35 topics). Given a set of all relevant documents for one topic, we find a clustering of this set at every possible threshold using the star algorithm. A document from the test set is relevant if its similarity to at least one cluster center is at least the threshold used to obtain the clustering. We compare the set of documents selected this way to the set of relevant test documents using the F measure, and find the threshold that maximizes F. The best F measure and the corresponding threshold are recorded. We also compute the mean similarity and the standard deviation of the set of training relevant documents. The difference between the best threshold and the mean similarity is expressed in terms of standard deviation and plotted. The plot in Figure 3 shows the number of topics for which the best threshold is within a certain distance from the mean. We observed that most topics are within one standard deviation from the mean and used this bound as our empirical threshold.

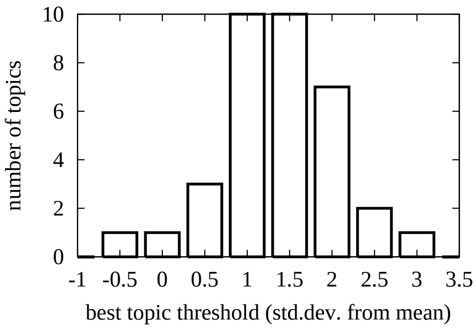

Figure 3: This figure shows the distribution of topics in terms of standard deviations from the mean threshold.

The empirical results in Figure 3 provide the threshold used in a filtering system: the mean similarity plus one standard deviation with respect to the training relevant set.

retrieved relevant documents does not matter, as long as they fit into a small number of topics.

4.3.4 Experiments

We have used the above collections to evaluate the performance of cluster-based filtering. In addition to the filtering algorithm based on the star clustering described in Section 4.2 we tested the following simpler filtering strategies.

Filtering around the topic centroid. This method is similar to the regular document retrieval around a query, and assumes that relevant documents are closer to the topic description than non-relevant ones. We take a set consisting of all training relevant documents for a topic, find the centroid of this set, and the minimum similarity between the centroid and vectors in the set. We select documents from the test set which similarity to the centroid is no less than the minimum similarity established on the training set. Compare the selected set of documents to the set of relevant test documents using $F = 2pr/(p + r)$ measure. We will refer to this method by $CENT$, and look at two variations. In the first variation called *static*, we use the centroid and the best threshold from the training collection throughout the filtering experiments. In the second variation, called *moving*, we adjust the centroid with each new relevant document added.

Filtering using the optimal cluster in a topic. This method considers a possibility of uneven distribution of relevant documents around the topic, and attempts to find the area most populated with relevant documents. We take a set consisting of all training relevant documents for a topic, for every possible threshold find all clusters in this set using the star algorithm. A document from the test set is selected if its similarity to a cluster center is no less than the threshold used to obtain the clustering. Compare the selected set of documents to the set of relevant test documents using $F = 2pr/(p + r)$ measure, find the threshold that maximizes F. We will refer to this method by $CLUS$, and look at two variations. In the first variation called *static*, we use the centroid of the best cluster from the training collection throughout the filtering experiments. In the second variation, called *moving*, we adjust the centroid with each new relevant document added.

Filtering using the star algorithm. Finally, we use the filtering algorithm outlined in Section 4.2. We apply this algorithm using a fixed pre-filtering threshold and a range of clustering thresholds. We compare the retrieved set of documents to the set of relevant test documents using $F = 2pr/(p+r)$ measure, and find the threshold that maximize F. We will refer to this method by $STAR$.

Figure 2 presents the results from the five experiments outlined above on the three test collections. We describe the performance of each algorithm as the average F-measure, where F is averaged over all topics. Thus, the higher the F value, the better the performance. We observe that the moving methods outperform the static methods and that the cluster methods outperform the centroid methods. Furthermore, the method based on the star algorithm has the best performance. Thus, we conclude that clustering, and especially the star algorithm are useful in improving the performance of the filtering task.

	static CENT	moving CENT	static CLUS	moving CLUS	STAR
FBIS	.230	.263	.243	.285	.294
AP	.257	.285	.260	.302	.310
News	.842	.893	.904	.907	.907

Table 2: This tables shows the best F_{avg} achieved by each of the three filtering algorithms described above (with two variations for the centroid (CENT) and cluster (CLUS) algorithms) over several filtering thresholds.

5 Discussion

We have described an applications of the star information organization system to filtering. While exploring the cluster-based paradigm to these applications we found the following advantages over non cluster-based approaches. (1) Clustered organization makes initial topic exploration and query refinement easy. (2) The filtering is recall-oriented, *i.e.*, when a document status is uncertain, the document is presumed relevant and retrieved. This lowers the chance of missing a relevant document; at the same time the clustered organization diminishes the impact on the user of inevitably higher volume of non-relevant documents in the retrieved set. (3) The clustered view and high recall provide an excellent and easy opportunity for the user to explore and change the topic. (4) To adjust topic definition the user needs to provide relevance feedback only for the star cluster centers, instead of all retrieved documents. We believe that clustering provides a "content-level compression" that can have a large impact on information organization from the point of view of information transmission and also for dealing with information overload for users. In the future, we plan to continue these studies to measure this kind of compression. We hope that this work will lead to scalable approaches to information access.

Acknowledgements

This work is supported in part by ONR contract N00014-95-1-1204, Rome Labs contract F30602-98-C-0006, DARPA contract F30602-98-2-0107, NSF award BCS-9978116 and Air Force MURI contract F49620-97-1-0382.

References

[AB84] M. Aldenderfer and R. Blashfield, *Cluster Analysis*, Sage, Beverly Hills, 1984.

[APL98] J. Allan, R. Papka, and V. Lavrenko. On-line New Event Detection and Tracking. In *Proceedings of SIGIR*, 1998.

[JPR98] J. Aslam, K. Pelekhov, and D. Rus, Static and dynamic information organization using star clusters, in *Proceedings of the 1998 Conference on Information Knowledge Management*, Baltimore, MD 1998.

[JPR99] J. Aslam, K. Pelekhov, and D. Rus, A practical clustering algorithm for information organization, In *Proceedings of the 1999 Symposium on Discrete Algorithms*, Baltimore, MD 1999.

[JRR00] J. Aslam, F. Reiss, and D. Rus, Scalability studies in information organization In *Proceedings of RIAO 2000*, Paris, France, 2000.

[Bur95] R. Burgin, The retrieval effectiveness of five clustering algorithms as a function of indexing exhaustively, *Journal of American Society for Information Science* 46(8):562-572, 1995.

[Cal96] J. Callan. Document Filtering With Inference Networks. In *Proceedings of the 19th International Conference on Research and Development in Information Retrieval* (SIGIR 96), Zurich, Switzerland, pp. 262-269, 1996.

[Cal98] J. Callan. Learning While Filtering Documents. In *Proceedings of SIGIR 98*, Melbourne, Australia, 1998.

[Can93] F. Can, Incremental clustering for dynamic information processing, in *ACM Transactions on Information Systems*, no. 11, pp143-164, 1993.

[CFG99] U. Cetintemel, M. Franklin, and L. Giles. Flexible user profiles for large scale data delivery. Technical report CS-TR-4005, Computer Science Department, University of Maryland, 1999.

[CCFM97] M. Charikar, C. Chekuri, T. Feder, and R. Motwani, Incremental clustering and dynamic information retrieval, in *Proceedings of the* 29^{th} *Symposium on Theory of Computing*, 1997.

[Cro77] W. B. Croft. Clustering large files of documents using the single-link method. *Journal of the American Society for Information Science*, pp189-195, November 1977.

[CKP93] D. Cutting, D. Karger, and J. Pedersen. Constant interaction-time scatter/gather browsing of very large document collections. In *Proceedings of the* 16^{th} *SIGIR*, 1993.

[ERS98] D. Eichmann, M. Ruiz, and P. Srinivasan. Cluster-Based Adaptive and Batch Filtering. In *Proceedings of the Seventh Text REtrieval Conference (TREC-7)*, 1998.

[FO95] C. Faloutsos and D. Oard. A Survey of Information Retrieval and Filtering Methods. Technical Report CS-TR-3514, Department of Computer Science, University of Maryland, 1995.

[Hul97] D. A. Hull, The TREC-6 Filtering Track: Description and Analysis, *The Seventh Text REtrieval Conference (TREC-6)*, 1997.

[Hul98] D. A. Hull, The TREC-7 Filtering Track: Description and Analysis, *The Seventh Text REtrieval Conference (TREC-7)*, 1998.

[HPS96] D. A. Hull, J. O. Pedersen, and H. Schütze. Method Combination for Document Filtering. In *Proceedings of the 1996 International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 279–287, 1996.

[JR71] N. Jardine and C.J. van Rijsbergen. The use of hierarchical clustering in information retrieval, 7:217-240, 1971.

[Lew95] D. Lewis. The TREC-5 filtering track. In eds. E. Voorhees and D. Harman. *The Fifth Text REtrieval Conference*, pp75–96. NIST Special Publication 500-238.

[LY94] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *Journal of the ACM* 41, 960–981, 1994.

[MMLP97] J. Mostafa, S. Mukhopadhyay, W. Lam, and M. Palakal. A multilevel approach to intelligent information filtering: model, system and evaluation. *ACM Transactions on Information Systems*, vol. 15, no. 4, pp 368–399, 1997.

[PA98] R. Papka and J. Allan. On-Line New Event Detection using Single Pass Clustering. UMass Computer Science Technical Report TR98-21, 1998.

[Rij79] C.J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 1979.

[Sal91] G. Salton. The Smart document retrieval project. In *Proceedings of the Fourteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 356-358.

[SHP95] H. Schütze, D. Hull, and J. Pedersen. A comparison of classifiers and document representations for the routing problem. In *Proceedings of SIGIR*, 1995.

[Sha86] W. Shaw, On the foundation of evaluation, *Journal of the American Society for Information Science*, 37, 346-348, 1986.

[SBH97] W. Shaw Jr., R. Burgin and P. Howell. Performance Standards and Evaluations in IR Test Collections: Cluster-Based Retrieval Models. *Information Processing & Management* 33(1) 1–14, 1997.

[Sib73] R. Sibson. SLINK: an optimally efficient algorithm for the single link cluster method. *Computer Journal* 16, pp30–34, 1973.

[Voo85] E. Voorhees. The cluster hypothesis revisited. In *Proceedings of the* 8^{th} *SIGIR*, pp 95-104, 1985.

[Wil88] P. Willett. Recent trends in hierarchical document clustering: A critical review. *Information Processing and Management*, 24:(5):577-597, 1988.

@InProceedingsyan:sift, author = Tak W. Yan and Hector Garcia-Molina, title = "SIFT - a tool for wide-area information dissemination", booktitle = "Proceedings of the 1995 Usenix Technical Conference", year = 1995, pages = 177-186,

[YG95] T. W. Yan and H. Garcia-Molina. SIFT - a tool for wide-area information dissemination". In *Proceedings of the 1995 Usenix Technical Conference*, pp 177–186, 1995.

[Zuc93] D. Zuckerman. NP-complete problems have a version that's hard to approximate. In *Proceedings of the Eight Annual Structure in Complexity Theory Conference*, IEEE Computer Society, 305–312, 1993.